图文 015、大厂面试题：什么情况下JVM内存中的一个对象会被垃圾回

详情 评论

大厂面试题

什么情况下JVM内存中的一个对象会被垃圾回收?

给大家推荐一套质量极高的Java面试训练营课程：

作者是中华石杉，石杉老哥是我之前所在团队的 Leader，骨灰级的技术神牛！

大家可以点击下方链接，了解更多详情，并进行试听：

21天互联网Java进阶面试训练营（分布式篇）

重要说明：

最近不少同学留言反馈，说希望建立一个微信群，供大家进行JVM专栏的学习交流。

这个提议非常好，不过管理微信群是一件挺费时的事儿，我平时工作较忙，实在抽不出时间来进行群管理.

正好石杉老哥的面试训练营建了微信交流群，并且还请了不少一线大厂的助教。

因此跟石杉老哥商量了一下，决定厚着脸皮“鸠占鹊巢”。购买了我JVM专栏的小伙伴，可以加入石杉老哥的微信群，在群里讨论交流技术。

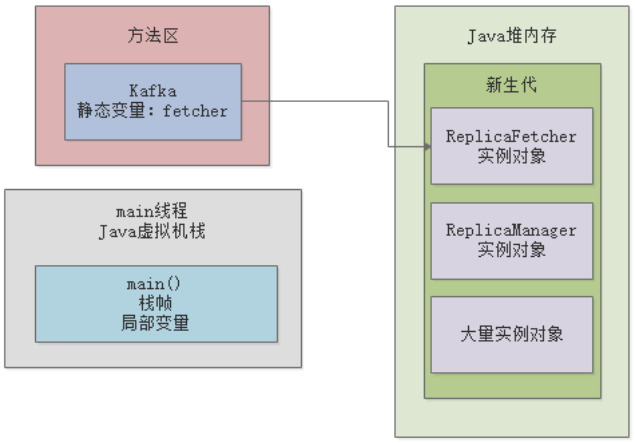
如何加群，请参见文末（注：如果之前已经加过的，就不要重复加群了）

推荐结束，正文开始

目录：

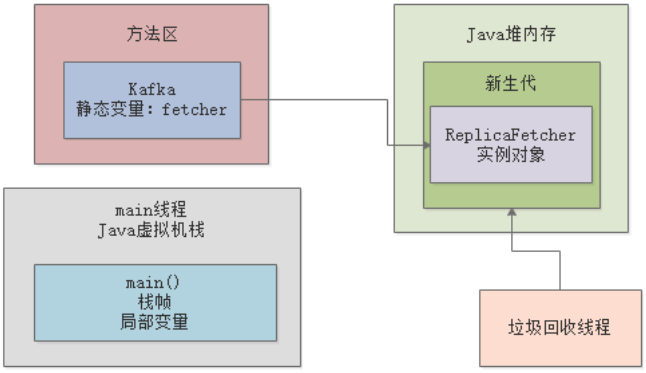
1、什么时候会触发垃圾回收?

通过之前的学习，相信大家都知道一点，平时我们系统运行创建的对象都是优先分配在新生代里的，如下图所示。



然后如果新生代里的对象越来越多，都快满了，此时就会触发垃圾回收，把新生代没有人引用的对象给回收掉，释放内存空间

大家务必注意，这就是新生代一个核心的垃圾回收触发时机，如下图。



那么本文就来针对这个过程，再次梳理其中的一些细节，看看触发垃圾回收的时候，到底是按照一个什么样的规则来回收垃圾对象的。

2、被哪些变量引用的对象是不能回收的?

首先第一个问题，一旦新生代快满了，那么垃圾回收的时候，到底哪些对象是能回收的，哪些对象是不能回收的呢?

这个问题非常好解释，JVM中使用了一种可达性分析算法来判定哪些对象是可以被回收的，哪些对象是不可以被回收的。

这个算法的意思，就是说对每个对象，都分析一下有谁在引用他，然后一层一层往上去判断，看是否有一个GC Roots。

这句话相当的抽象，是不是?

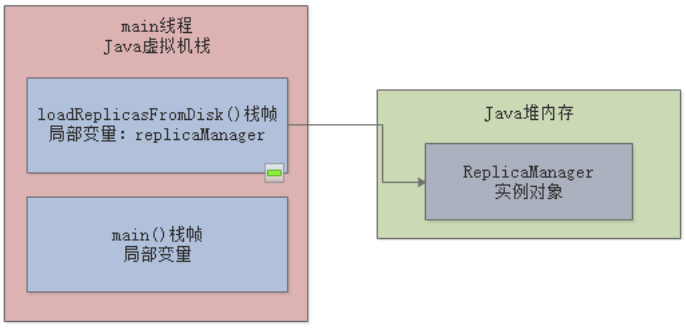
没关系，我们的特点就是一步一图，保证你看明白。

比如最常见的，就是下面的一种情况。

```
public class Kafka {
    public static void main(String[] args) {
        loadReplicasFromDisk();
    }
    public static void loadReplicasFromDisk() {
        ReplicaManager replicaManager = new ReplicaManager();
    }
}
```

上面的代码其实就是在一个方法中创建了一个对象，然后有一个局部变量引用了这个对象，这种情况是最常见的

此时如下图所示。“main()”方法的栈帧入栈，然后调用“loadReplicasFromDisk()”方法，栈帧入栈，接着让局部变量“replicaManager”引用堆内存里的“ReplicaManager”实例对象。



假设现在上图中“ReplicaManager”对象被局部变量给引用了，那么此时一旦新生代快满了，发生垃圾回收，会去分析这个“ReplicaManager”对象的可达性

这时，发现他是不能被回收的，因为他被人引用了，而且是被局部变量“replicaManager”引用的。

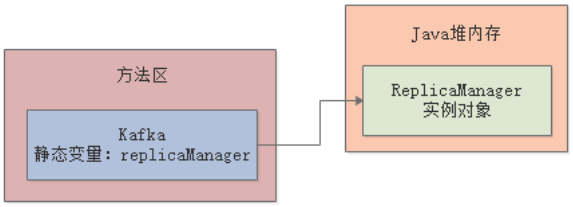
在JVM规范中，局部变量就是可以作为GC Roots的

只要一个对象被局部变量引用了，那么就说明他有一个GC Roots，此时就不能被回收了。

另外比较常见的一个情况，其实就是类似下面的代码。

```
public class Kafka {
    public static ReplicaManager replicaManager = new ReplicaManager();
}
```

大家可以分析一下上面的代码，如下图所示。



大家按照上图思考一下，此时垃圾回收的时候一分析，发现这个“ReplicaManager”对象被Kafka类的一个静态变量“replicaManager”给引用了

此时在JVM的规范里，静态变量也可以看做是一种GC Roots，此时只要一个对象被GC Roots引用了，就不会去回收他。

所以说，**一句话总结**：只要你的对象被方法的局部变量、类的静态变量给引用了，就不会回收他们。

3、Java中对象不同的引用类型

关于引用和垃圾回收的关系，大家在这里务必有脑子里要引入一个新的概念，那就是Java里有不同的引用类型。

分别是强引用、软引用、弱引用和虚引用。下面分别用代码来示范一下。

强引用，就是类似下面的代码：

```
public class Kafka {
    public static ReplicaManager replicaManager = new ReplicaManager();
}
```

这个就是最普通的代码，一个变量引用一个对象，只要是强引用的类型，那么垃圾回收的时候绝对不会去回收这个对象的。

接着是软引用，类似下面的代码。

```
public class Kafka {
    public static SoftReference<ReplicaManager> replicaManager =
            new SoftReference<ReplicaManager>(new ReplicaManager());
}
```

就是把“ReplicaManager”实例对象用一个“SoftReference”软引用类型的对象给包裹起来了，此时这个“replicaManager”变量对“ReplicaManager”对象的引用就是软引用了。

正常情况下垃圾回收是不会回收软引用对象的，但是如果你进行垃圾回收之后，发现内存空间还是不够存放新的对象，内存都快溢出了

此时就会把这些软引用对象给回收掉，哪怕他被变量引用了，但是因为他是软引用，所以还是要回收。

接着是弱引用，类似下面的代码。

```
public class Kafka {
    public static WeakReference<ReplicaManager> replicaManager =
            new WeakReference<ReplicaManager>(new ReplicaManager());
}
```

这个其实非常好解释，你这个弱引用就跟没引用是类似的，如果发生垃圾回收，就会把这个对象回收掉。

虚引用，这个大家其实暂时忽略他也行，因为很少用。

其实这里比较常用的，就是**强引用和软引用**，强引用就是代表绝对不能回收的对象，软引用就是说有的对象可有可无，如果内存实在不够了，可以回收他。

4、finalize()方法的作用

现在大家理解完了GC Roots和引用类型的概念，基本都知道了，哪些对象可以回收，哪些对象不能回收。

有GC Roots引用的对象不能回收，没有GC Roots引用的对象可以回收，如果有GC Roots引用，但是如果是软引用或者弱引用的，也有可能被回收掉。

接着就是到回收的环节了，假设没有GC Roots引用的对象，是一定立马被回收吗?

其实不是的，这里有一个finalize()方法可以拯救他自己，看下面的代码。

```
public class ReplicaManager {
     public static ReplicaManager instance;
     @Override
     protected void finalize() throws Throwable {
          ReplicaManager.instance = this;
     }
}
```

假设有一个ReplicaManager对象要被垃圾回收了，那么假如这个对象重写了Object类中的finialize()方法

此时会先尝试调用一下他的finalize()方法，看是否把自己这个实例对象给了某个GC Roots变量，比如说代码中就给了ReplicaManager类的静态变量。

如果重新让某个GC Roots变量引用了自己，那么就不用被垃圾回收了。

不过说实话，这个东西没必要过多解读，因为其实平时很少用，就是给大家梳理出来这些细节，让大家清楚而已。

5、昨日思考题

上周的思考题和作业是一个意思，就是让大家去思考，自己负责的系统的内存压力，然后就是JVM内存大小是否合理，如果业务暴增100倍，是否会有内存问题。

作业更加详细的提示大家，自己画出核心业务流程图，然后一点点去分析，这是一个非常重要的技能。

其实JVM实战技能里的第一步，就是合理预估系统内存压力，合理设置JVM内存大小。

6、今日思考题

思考下面的代码。

```
public class Kafka {
     public static ReplicaManager replicaManager = new ReplicaManager();
}
public class ReplicaManager {
     public ReplicaFetcher replicaFetcher = new ReplicaFetcher();
}
```

上述代码下，如果垃圾回收，会回收ReplicaFetcher对象吗？为什么？

End

如何加群?

1、添加微信号：Giotto1245 （微信名：Jarvis）

2、发送 Jvm专栏的购买截图

3、人工操作，发送截图后请耐心等待被拉群

最后提醒：之前加过面试群的同学就不要重复加了

常见问题解答：

一、 如何生成自己的分享海报并获取返现?

方式1：

点击文章右上角**邀请好友**（如下图），生成自己的专属海报。

将海报发送给好友或分享朋友圈，朋友通过扫描你分享的海报购买课程，你将**获取返现24元**，可在个人中心中提现：

累计邀请30人，你将升级为高级推广员，此后每成功邀请一位朋友，返现翻倍。换句话说，从第31人开始，每成功邀请一位朋友，你将**获取返现48元**